Prof. Dr.-Ing. Ralf Steinmetz
Multimedia communications Lab

Dr. Florian Mehm
Dipl. Inf. Robert Konrad

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Game Technology
Winter Semester 2016/2017

Solution 8

General Information

- The exercises may be solved by teams of up to three people.
- The solutions have to be uploaded to the Git repositories assigned to the individual teams.
- **The submission date (for practical and theoretical tasks) is noted on top of each exercise sheet.**
- If you have questions about the exercises write a mail to game-technology@kom.tu-darmstadt.de or use the forum at https://www.fachschaft.informatik.tu-darmstadt.de/forum/viewforum.php?f=557

P8. Practical Tasks: Physics (5 Points)

In this exercise, the overall task is to build a demo in which you can shoot spheres from the camera position by pressing the space key and have them interact with each other and a plane. The code is provided for you; your task is to fill out the respective functions. The code can be found at https://github.com/TUDGameTechnology/Exercise7.git

Please remember to push into a branch called "exercise8".

You can find the practical solutions at https://github.com/TUDGameTechnology/Solution8.git.

P8.1 Particle Systems – Billboards (1 point)

You can see in the exercise code that the particles are being spawned, but when the camera turns, they are visible from the side and back. Instead, we want them oriented towards the camera. Implement this rotation. The view matrix of the camera is available to the particle system.

P8.2 Particle Systems – Control parameters (2 points)

Decide on an effect you want to create using particle systems that uses one control parameter that is not present in the code you are given. For example, you can implement the "fire" example from the slides. Other control parameter could be the rotation of the particle billboards (you might want to change the texture in this case as the provided texture is symmetrical), the size of the billboards or the mass. For rain, you could spawn a "splash" particle when the rain hits the ground.

P8.3 Numerical Integration (1 point)

Implement an Euler integrator for the physics computations. The necessary function can be found in "PhysicsObject.cpp". Note that it is beneficial to multiply the resulting velocity during the integration step with a damping coefficient. This coefficient accounts for energy that is normally lost when objects move and interact, and helps the system come to rest eventually.

P8.4 Sphere-Sphere Intersections (1 point)

Implement an intersection test for two spheres with each other. This task entails implementing the functions
bool IntersectsWith(const SphereCollider& other);
vec3 GetCollisionNormal(const SphereCollider& other);
float PenetrationDepth(const SphereCollider& other);

which are found in "Collision.h"

T8. Theoretical Tasks: Physics (5 Points)

T8.1 Integration constant acceleration (1.5 Points)

Imagine a sphere with mass m=1kg that is falling down in a vacuum. At t=0, the sphere is released. The velocity v at t=0 is 0. The acceleration a is constant at 10 m/s² in the negative y-direction.

Calculate the movement of the sphere by integrating the equations of motion using the Euler method as explained in the lecture (you can use a spreadsheet application or a script).

a) Use a time step of deltaT=1s and provide the values of height (z) and velocity (v) for the situation at time t=3s.
b) Use a time step of deltaT = 0.5s and provide the values. Compare the results and discuss them.

a)

t	y	v	f	gravity	acceleration	mass	deltaT
0	0	0	0	-10	-10	1	1
1	-10	-10	0	-10	-10	1	1
2	-30	-20	0	-10	-10	1	1
3	**-60**	**-30**	**0**	**-10**	**-10**	**1**	**1**
4	-100	-40	0	-10	-10	1	1
5	-150	-50	0	-10	-10	1	1
6	-210	-60	0	-10	-10	1	1
7	-280	-70	0	-10	-10	1	1
8	-360	-80	0	-10	-10	1	1
9	-450	-90	0	-10	-10	1	1
10	-550	-100	0	-10	-10	1	1
11	-660	-110	0	-10	-10	1	1
12	-780	-120	0	-10	-10	1	1

b)

t	y	v	f	gravity	acceleration	mass	deltaT
0	0	0	0	-10	-10	1	0,5
0,5	-2,5	-5	0	-10	-10	1	0,5
1	-7,5	-10	0	-10	-10	1	0,5
1,5	-15	-15	0	-10	-10	1	0,5
2	-25	-20	0	-10	-10	1	0,5
2,5	-37,5	-25	0	-10	-10	1	0,5
3	**-52,5**	**-30**	**0**	**-10**	**-10**	**1**	**0,5**
3,5	-70	-35	0	-10	-10	1	0,5
4	-90	-40	0	-10	-10	1	0,5
4,5	-112,5	-45	0	-10	-10	1	0,5
5	-137,5	-50	0	-10	-10	1	0,5
5,5	-165	-55	0	-10	-10	1	0,5
6	-195	-60	0	-10	-10	1	0,5

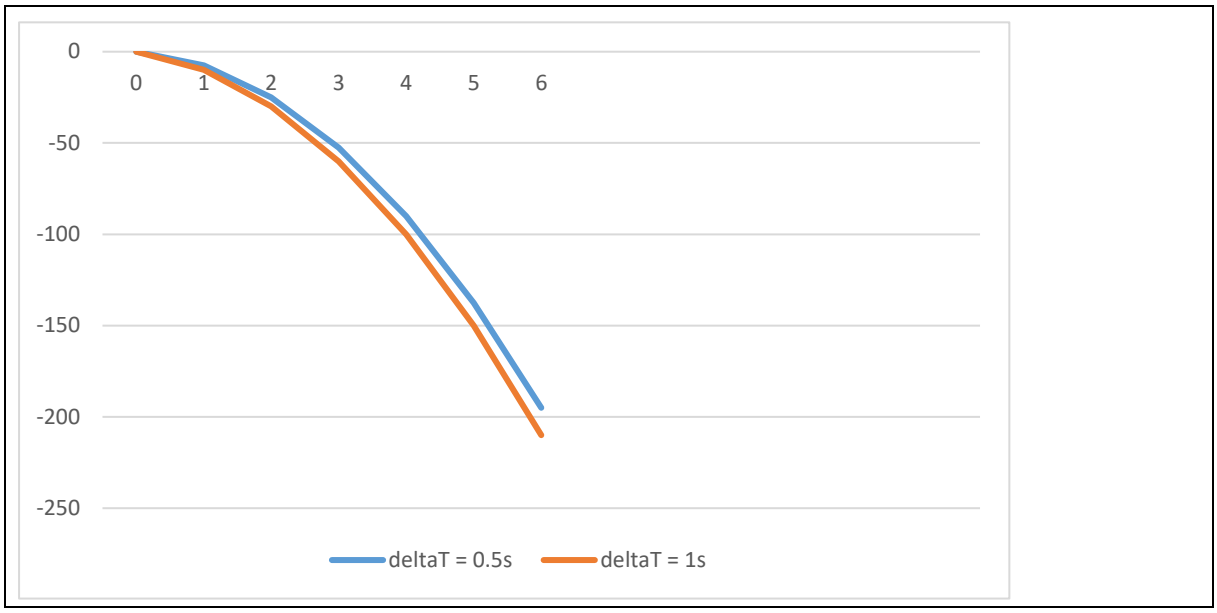
We can find an analytical solution by the formula

$$y(t) = v_0 t + \frac{1}{2} a t^2$$

Therefore,

$$y(3) = 0 \cdot 3 + \frac{1}{2} 10 \cdot 3^2$$
$$= \frac{1}{2} 10 \cdot 9 = 45$$

We see that the shorter time step could better approximate the correct value.



T8.2 Integration under drag (1.5 Points)

The sphere of task 2.1 is now dropped in the earth's atmosphere. Now, drag is acting on the object.

Drag is a force that acts on an object based on the velocity of the object. The faster the object, the more drag it experiences. We can approximate drag with the following formula:

$F_{drag} = -v_{normalized} (k_1 |v| + k_2 |v|^2)$

The result is a force that acts in the opposite direction of the object's movement (indicated by $-v_{normalized}$, the normal vector in the direction of movement). The amount of drag results from the coefficients k1 and k2. Choose k1 = k2 = 0.1.

Calculate the movement of the sphere by integrating the equations of motion using the Euler method as explained in the lecture (you can use a spreadsheet application or script). Make sure to add the acceleration caused by the drag in the formulas you use.

a) Use a time step of deltaT=1s and provide the values of height (z) and velocity (v) for the situation at time t=2s.
b) Use a time step of deltaT = 0.2s and provide the values. Compare the results and discuss any differences.

Note: The values of k1 and k2 are not chosen realistically. For a real object, the constants depend on the cross-section of the object that is oriented in the direction of travel, the drag coefficient of the object as well as the density and viscosity of the medium the object travels through. As long as the simulated object falls down and is not pushed up by drag forces, your calculation is probably correct.

a)

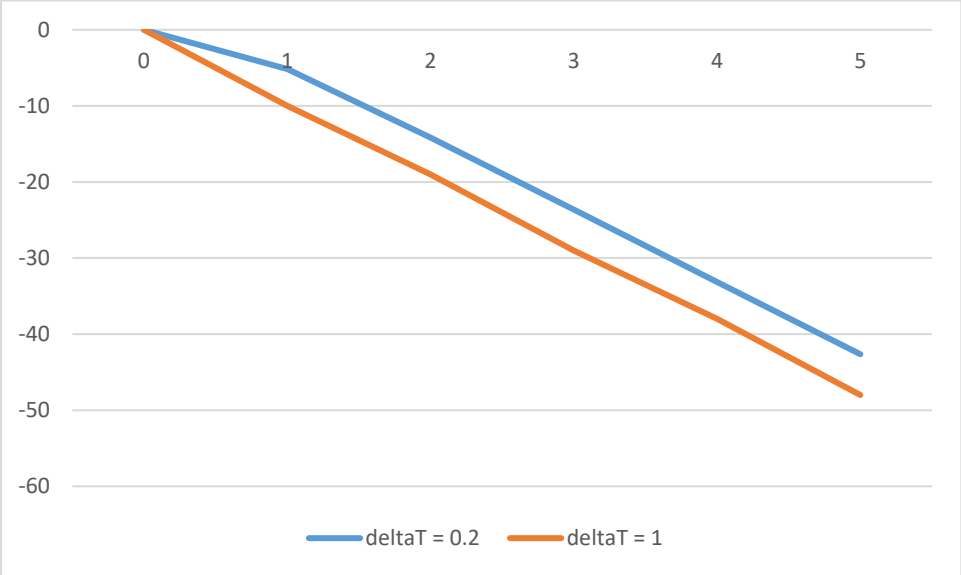
t	y	v	f	gravity	acc	mass	k1	k2	deltaT
0	0	0	0	-10	-10	1	0,1	0,1	1
1	-10	-10	11	-10	1	1	0,1	0,1	1
2	**-19**	**-9**	**9**	**-10**	**-1**	**1**	**0,1**	**0,1**	**1**
3	-29	-10	11	-10	1	1	0,1	0,1	1
4	-38	-9	9	-10	-1	1	0,1	0,1	1
5	-48	-10	11	-10	1	1	0,1	0,1	1

6	-57	-9	9	-10	-1	1	0,1	0,1	1
7	-67	-10	11	-10	1	1	0,1	0,1	1
8	-76	-9	9	-10	-1	1	0,1	0,1	1
9	-86	-10	11	-10	1	1	0,1	0,1	1
10	-95	-9	9	-10	-1	1	0,1	0,1	1

b)

t	y	v	f	gravity	acc	mass	k1	k2	deltaT
0	0	0	0	-10	-10	1	0,1	0,1	0,2
0,2	-0,4	-2	0,6	-10	-9,4	1	0,1	0,1	0,2
0,4	-1,176	-3,88	1,89344	-10	-8,10656	1	0,1	0,1	0,2
0,6	-2,2762624	-5,501312	3,57657457	-10	-6,42342543	1	0,1	0,1	0,2
0,8	-3,63346182	-6,78599709	5,28357535	-10	-4,71642465	1	0,1	0,1	0,2
1	-5,17931822	-7,72928201	6,74710825	-10	-3,25289175	1	0,1	0,1	0,2
1,2	-6,85529029	-8,37986037	7,86019201	-10	-2,13980799	1	0,1	0,1	0,2
1,4	-8,61685469	-8,80782196	8,63855497	-10	-1,36144503	1	0,1	0,1	0,2
1,6	-10,4328769	-9,08011097	9,15285262	-10	-0,84714738	1	0,1	0,1	0,2
1,8	-12,282785	-9,24954045	9,48035389	-10	-0,51964611	1	0,1	0,1	0,2
2	**-14,1534789**	**-9,35346967**	**9,68408645**	**-10**	**-0,31591355**	**1**	**0,1**	**0,1**	**0,2**
2,2	-16,0368094	-9,41665238	9,80899944	-10	-0,19100056	1	0,1	0,1	0,2
2,4	-17,9277799	-9,45485249	9,88490881	-10	-0,11509119	1	0,1	0,1	0,2
2,6	-19,823354	-9,47787073	9,93079043	-10	-0,06920957	1	0,1	0,1	0,2
2,8	-21,7216966	-9,49171264	9,95843215	-10	-0,04156785	1	0,1	0,1	0,2
3	-23,6217018	-9,50002621	9,97505242	-10	-0,02494758	1	0,1	0,1	0,2

3,2	-25,5227049	-9,50501573	9,98503397	-10	-0,01496603	1	0,1	0,1	0,2
3,4	-27,4243067	-9,50800893	9,99102428	-10	-0,00897572	1	0,1	0,1	0,2
3,6	-29,3262675	-9,50980408	9,99461777	-10	-0,00538223	1	0,1	0,1	0,2
3,8	-31,2284436	-9,51088052	9,99677289	-10	-0,00322711	1	0,1	0,1	0,2
4	-33,1307488	-9,51152595	9,99806518	-10	-0,00193482	1	0,1	0,1	0,2
4,2	-35,0331314	-9,51191291	9,99884001	-10	-0,00115999	1	0,1	0,1	0,2
4,4	-36,9355604	-9,51214491	9,99930457	-10	-0,00069543	1	0,1	0,1	0,2
4,6	-38,8380172	-9,51228399	9,99958308	-10	-0,00041692	1	0,1	0,1	0,2
4,8	-40,7404907	-9,51236738	9,99975005	-10	-0,00024995	1	0,1	0,1	0,2
5	-42,6429741	-9,51241737	9,99985016	-10	-0,00014984	1	0,1	0,1	0,2



Note that the values of k1 and k2 are arbitrary and not physically based. Still, the object reaches a terminal velocity of about 9.5 units per second. When it reaches this velocity, the fall becomes a linear movement.

For the chosen situations, the different update rates of the physics are not incurring a large difference. However, depending on what the gameplay and other requirements are, already these differences could be substantial. For example, if we need a deterministic physics simulation, e.g. for a lockstep multiplayer game, it would be important to choose the same step size to have exactly the same results.

T8.3 Sphere-Plane-Collision (2 points)

Consider the situation below, which gives you information about the current state of a sphere and an immovable plane. You can disregard gravity for this exercise.

Plane: $\boldsymbol{d} = \mathbf{1}, \boldsymbol{n} = \begin{pmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{pmatrix}$

Circle: $\text{radius} = \mathbf{3}$, $\text{position} = \begin{pmatrix} \mathbf{5} \\ \mathbf{2} \\ \mathbf{1} \end{pmatrix}$, $\text{velocity} = \begin{pmatrix} -\mathbf{0.5} \\ \mathbf{1} \\ \mathbf{1} \end{pmatrix}$

a) Calculate the following values. Please include your calculations.

Distance (sphere center to plane):

Collision normal:

Penetration depth:

Separating Velocity:

Distance: *We know that we can get the distance of a point to a plane by entering the point into the equation of the plane:*

$$\begin{aligned} D &= xn - d \\ &= \begin{pmatrix} \mathbf{5} \\ \mathbf{2} \\ \mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{pmatrix} - 1 \\ &= 2 - 1 \\ &= 1 \end{aligned}$$

Collision normal: *The collision normal is the same as the normal of the plane:*

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{pmatrix}$$

Penetration depth: *The penetration depth is the part of the sphere that is inside the plane. We get this by comparing the distance of the sphere center to the plane and the sphere radius:*

$$r - d = 3 - 1 = 2$$

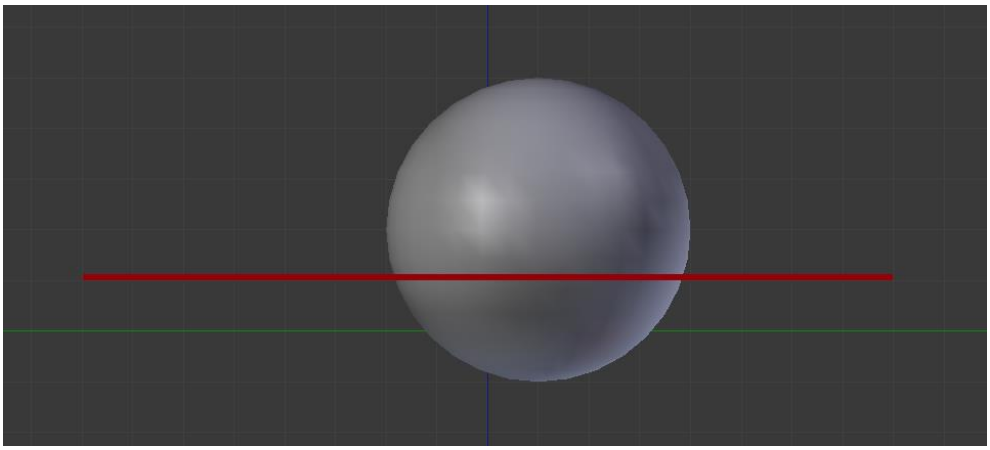
Separating velocity: *To get the separating velocity, we need to consider the projection of the sphere's velocity onto the collision normal. We do not need to use two velocities since the plane is not moving.*

$$\begin{pmatrix} -\mathbf{0.5} \\ \mathbf{1} \\ \mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{pmatrix} = 1$$

b) Are the sphere and the plane colliding? Explain your answer.

Yes, they are. We can read this from the penetration depth: It is positive, therefore, the sphere and the plane are intersecting.

You can also see this in the 3D rendering of the scene:



c) Should we apply some collision response? Explain your answer. (You don't need to specify which collision response, if any is required).

Since the separating velocity is > 0, the objects are already separating, so we don't need to separate them.

We might reset the sphere to be outside of the plane immediately, as described in the lecture slides on slides 45 – 46.